**IN THE CLAIMS:**

Please cancel claim 15 without prejudice or disclaimer as to the subject matter contained therein.

Please amend the claims as shown in the following claims listing.

1. (Currently amended) A method comprising:

a service processor executing application software for configuring a computer system into one or more domains and for performing diagnostics;

providing program instructions ~~that describe a sequence of one or more transactions for manipulating hardware components of a system~~ comprising operational descriptions that invoke one or more hardware access code segments that provide, to said program instructions, one or more specific transaction sequences to access and manipulate one or more respective registers within a system processor and a client;

s~~aid program instructions calling one or more code segments that include specific information associated with particular hardware components of said system and wherein said program instructions are independent of said specific information;~~

wherein said program instructions comprise a first level of hardware abstraction that hides specific hardware functionality from the application software;

wherein said one or more hardware access code segments comprise a second level of hardware abstraction that hides specific register access information from said program instructions;

translating said program instructions into an executable form; and

executing said executable form of said program instructions to manipulate said hardware components of said system from one consistent state to a next consistent state in response to execution of said application software.

2. (Currently amended) The method as recited in claim 1, wherein to access and manipulate one or more respective registers ~~said manipulating hardware components~~ includes reading and writing information within [[a]] one or more control ~~register~~ registers.

3. (Currently amended) The method as recited in claim 1, wherein said ~~hardware components~~ system processor and said client each include one or more state machine implementations each including a plurality of states and wherein said plurality of states are effected in response to invocation of said one or more code segments.

4. (Currently amended) The method as recited in claim 1, wherein said specific register access information includes control register-specific information.

5. (Original) The method as recited in claim 4, wherein said control register-specific information includes a control register name.

6. (Original) The method as recited in claim 5 further comprising specifying said control register name using an integer arithmetic expression that is evaluated as an integer value and translated into a character representation of said integer value.

7. (Original) The method as recited in claim 4, wherein said control register-specific information includes a control register field including one or more particular bits.

8. (Original) The method as recited in claim 7 further comprising specifying and writing a predetermined value to said control register field using a field assignment including a name of said control register field and an expression including a variable representative of said one or more particular bits being written.

9. (Currently amended) The method as recited in claim 1 further comprising deriving said specific register access information associated with particular hardware

components from an output of a design tool, wherein said output corresponds to a hardware definition language representation of said hardware components.

10. (Original) The method as recited in claim 1, wherein describing said sequence of one or more transactions for manipulating hardware components includes specifying an access operation on a given control register without defining a specific bus route for conveying said operation.

11. (Original) The method as recited in claim 1 further comprising storing a value in a storage variable during a transaction, wherein said value is accessible by an instruction executed subsequent to completion of said transaction.

12. (Original) The method as recited in claim 1, wherein said program instructions are written in a descriptive abstract programming language that is not directly compilable.

13. (Original) The method as recited in claim 1 further comprising deriving said specific information associated with particular hardware components from a computer information model (CIM) representation by associating each particular hardware component with a CIM class having a CIM class name and creating, for each CIM class, a corresponding unique type by appending "_cim_t" to said CIM class name.

14. (Currently amended) The method as recited in claim 1 further comprising specifying a hardware function programming interface type that corresponds to a particular partition of functionality within a given hardware component type, wherein two identically named hardware access code segments have substantially a same semantic effect and different implementations when called by two different hardware function programming interfaces.

15. (Cancelled)

16. (Currently amended) A computer system comprising:

a system processor configured to execute instructions associated with user software;

a client coupled to said system processor via a system interconnect; and

a service processor coupled to said system processor and to said client via a maintenance bus, wherein said service processor is configured to execute application software for configuring said computer system into one or more domains and for performing diagnostics;

wherein said service processor is further configured to:

execute ~~access~~ an executable form of program instructions for manipulating said system processor and said client from one consistent state to a next consistent state in response to execution of the application software;

wherein said program instructions comprise a first level of hardware abstraction that hides specific hardware functionality from the application software;

wherein said program instructions comprise operational descriptions that ~~describe a sequence of one or more transactions for manipulating said system processor and said client;~~

~~wherein said program instructions call~~ invoke one or more hardware access code segments ~~that include specific information associated with~~ that provide, to said program instructions, one or more specific transaction sequences to access and manipulate one or more respective registers within said system processor and said client ~~and wherein said program instructions are independent of said specific information~~; and

wherein said one or more hardware access code segments comprise a second level of hardware abstraction that hides specific register access information from said program instructions [[and]]

~~execute said executable form of said program instructions in response to executing said application software~~.

17. (Original) The computer system as recited in claim 16, wherein said system processor and said client each include one or more respective control registers, wherein said respective control registers include one or more bits configured to control and indicate an operating state of said system processor and said client.

18. (Original) The computer system as recited in claim 17, wherein said service processor is further configured to read and write information within said one or more respective control registers of said system processor and said client.

19. (Currently amended) The computer system as recited in claim 16, wherein said system processor and said client each include one or more state machine implementations each including a plurality of states and wherein said plurality of states are effected in response to invocation of said one or more hardware access code segments.

20. (Currently amended) The computer system as recited in claim 16, wherein said specific register access information includes control register-specific information.

21. (Original) The computer system as recited in claim 20, wherein said control register-specific information includes a control register name.

22. (Original) The computer system as recited in claim 21, wherein said program instructions include specifying said control register name using an integer arithmetic expression that is evaluated as an integer value and translated into a character representation of said integer value.

23. (Original) The computer system as recited in claim 21, wherein said control register-specific information includes a control register field including one or more particular bits.

24. (Original) The computer system as recited in claim 23, wherein said program instructions include specifying and writing a predetermined value to said control register

field using a field assignment including a name of said control register field and an expression including a variable representative of said one or more particular bits being written.

25. (Currently amended) The computer system as recited in claim 16, wherein said specific register access information associated with said system processor and said client is derived from an output of a design tool, wherein said output corresponds to a hardware definition language representation of said system processor and said client.

26. (Original) The computer system as recited in claim 16, wherein said program instructions include specifying an access operation on a control register without defining a specific bus route for conveying said operation.

27. (Original) The computer system as recited in claim 16, wherein said program instructions include storing a value in a storage variable during a transaction, wherein said value is accessible by an instruction executed subsequent to completion of said transaction.

28. (Original) The computer system as recited in claim 16, wherein said program instructions are written in a descriptive abstract programming language that is not directly compilable.

29. (Original) The computer system as recited in claim 16, wherein said specific information associated with particular hardware components is derived from a computer information model (CIM) representation by associating each particular hardware component with a CIM class having a CIM class name and creating, for each CIM class, a corresponding unique type by appending "_cim_t" to said CIM class name.

30. (Original) The computer system as recited in claim 16, wherein a hardware function programming interface type is specified that corresponds to a particular partition of functionality within a given hardware component type, wherein two identically named

code segments have a substantially same semantic effect and different implementations when called by two different hardware function programming interfaces.